AD-A073 904 NAVAL SURFACE WEAPONS CENTER DAHLGREN LAB VA F/G 9/2
TRICOMP6000 USER'S GUIDE.(U)
MAR 79 W CARUTHERS, H HUBER, J ZALOUDEK
UNCLASSIFIED NSWC/TR-3938 NL

1 OF 2

AD
A073904

END
DATE
FILMED
10-79
DDC

CONT.

FIED

1 OF 2

AD
A073904

MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AD A073904

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NSWC TR-3938	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) TRICOMP6000 USER'S GUIDE		5. TYPE OF REPORT & PERIOD COVERED FINAL
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) W. Caruthers H. Huber J. Zaloudek		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Surface Weapons Center (K54) Dahlgren, Virginia 22448		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE March 1979
		13. NUMBER OF PAGES 33
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

TRICOMP6000 is a compiler for the TRIDENT Higher Level Language (THLL) that runs on, and produces code for, the CDC 6700. This document describes the differences between TRICOMP6000 and TRICOMP which is the standard compiler that translates THLL programs into code for the TRIDENT Digital Control Computer (TDCC).

79 09 17 061

FOREWORD

The TRIDENT Higher Level Language (THLL) is used for fire control programming of the TRIDENT System, both for the Mk 98 FCS and the Mk 88 Mod 2 FCS. It is implemented by a compiler, TRICOMP, that runs on the CDC 6700 and produces code for the TRIDENT Digital Control Computer (TDCC).

A second compiler for THLL, TRICOMP6000, also runs on the CDC 6700 and translates THLL programs into CDC assembly code COMPASS. This compiler allows the use of THLL as a general purpose programming language for the CDC 6700. It is currently used primarily to maintain TRICOMP, which is written in THLL, and to develop the Data Reduction Software for the Verification and Evaluation System for TRIDENT (VEST).

The purpose of this document is to describe the differences between TRICOMP and TRICOMP6000 as far as the user has to know.

This document was prepared by the THLL Group, Fire Control Programming Branch (K54). Questions, comments, and suggestions should be directed to K54.

Released by

Ralph A. Niemann
RALPH A. NIEMANN, Head
Strategic Systems Department

Accession For
NTIS GRA&I ☑
DDC TAB □
Unannounced □
Justification________
By________
Distribution/
Availability Codes
Dist | Avail and/or special
A

CONTENTS

CONTENTS (CONT'D)

INTRODUCTION

TRICOMP6000 is a compiler for the TRIDENT Higher Level Language (THLL) that runs on, and produces code for, the CDC 6700. This document describes the differences between TRICOMP6000 and TRICOMP which is the standard compiler that translates THLL programs into code for the TRIDENT Digital Control Computer (TDCC).

This document is not self-contained. It is to be used in combination with the THLL User's Guide (Reference 1).

Differences between the two compilers are due to differences in the hardware of the CDC 6700 and the TDCC (60-bit word vs 32-bit word, one's complement vs two's complement, different instruction sets) and differences in the software environment (COMPASS, CDC Loader, and SCOPE vs TASS).

TWO COMPILER MODES

TRICOMP6000 has two distinct compiler modes, BP and CDC. The BP mode is used to maintain compatibility with TRICOMP which targets programs for the TDCC. CDC 6700 code is generated but an attempt is made to simulate TDCC behavior. Many exceptions exist which are noted in the following sections. The CDC mode uses the full word size of the CDC 6700 and removes some of the TDCC dependencies of the language. A comment in the compiler's mnemonic output indicates the compiler mode that was used.

BASIC ELEMENTS OF THLL

CHARACTER SET

Only the CDC character set is used. No attempt is made to produce TDCC printable characters.

OPERATORS

All of the THLL operators are implemented in TRICOMP6000. On the CDC 6700, a virtual address is equivalent to absolute address or physical address. The only difference between LOC and LOCA is the type of its result (Appendix B in the THLL User's Guide, Reference 1).

DELIMITERS

All of the THLL delimiters are implemented in TRICOMP6000, but some have little or no meaning in the environment of the CDC 6700. Examples: DOUBLE, EXEC, HALF, INTERRUPT, LINK, TASS.

IDENTIFIERS

Global and external names are truncated by the compiler to the first six characters; therefore, the names should be unique in the first six characters. If the name so obtained contains dots or is of the form AN, BN, or XN, where N=0,1,...,7, then the name is transformed to a unique seven-character symbol (page 2.1, Reference 2). These restrictions are imposed by the CDC loader and COMPASS assembler.

CONSTANTS

Bit Patterns vs Numbers

Since the CDC data are represented by different bit patterns and since the CDC arithmetic is in one's complement, the programmer should distinguish carefully between data such as numbers, strings, TRUE, FALSE, and bit patterns. For example, parts of data should not be extracted using components. In the best case, this will introduce a very sensitive dependency on the machine representation of data.

Binary, octal, and hexadecimal integers are considered to be bit patterns, and, with the exception of sign extension in BP mode, bit patterns have the same representation in both the CDC 6700 and TDCC. Decimal integers and real numbers are considered to be arithmetic and that value is represented by a machine-dependent bit pattern.

Integer and Double Constants

In BP mode, any decimal integer number that exceeds 31 bits of significance is considered to be a double value. The 32-bit integer value is sign-extended to fill a 60-bit CDC word. Double values, also, are placed in one 60-bit CDC word. An error message is emitted when a number or bit pattern exceeds 60 bits of significance. The error message is only a warning to note that the least significant 60`bits are used. Numbers and masks greater than 64 bits of significance are considered to be an error in TRICOMP and, at that point, TRICOMP6000 will perform identical error recovery (generate 0 of the appropriate type).

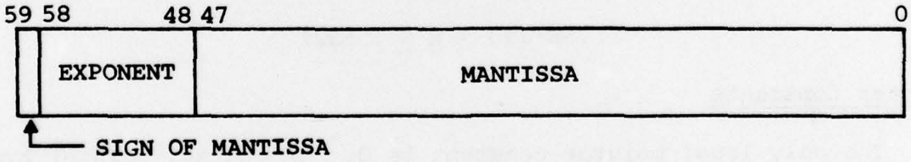
In BP mode, any bit pattern that exceeds 32 bits of significance is considered to be a double value. Sign extension of the most significant bit of an integer bit pattern will occur to fill the 60-bit CDC word. No adjustment is made for the difference between one's complement and two's complement arithmetic when bit patterns are sign extended. Care must be taken not to cause all 60 bits of a CDC word to be set. This negative 0 number is by many instructions interpreted as positive 0 (all bits off). The value TRUE can be used as a 60-bit single integer containing only the low-order 32 bits set. A 60-bit double value containing only the low-order 32 bits is obtained by using the constant X'0FFFFFFFF' (which specifies a bit pattern of 36 bits, and therefore double).

In CDC mode, any decimal integer or bit pattern up to 60 bits is considered to be an integer value.

Arithmetic Precision of HALF, INTEGER, DOUBLE. Since data of types HALF, INTEGER, and DOUBLE are all represented by 60-bit CDC integers, the precision for Halfs and Integers is higher on the CDC than on the TDCC while the precision for Doubles is less. There is no double arithmetic on the CDC and Integers and Doubles are treated alike. The arithmetic of the CDC 6700 will not divide, multiply, or float integers that exceed 48 bits of significance. Shifts, addition, subtraction, and bit operations work on 60-bit operands.

Real Constants

Real data are expressed as 60-bit floating point numbers having a mantissa of 48 bits plus sign and an exponent of 11 bits including sign. The CDC bit numbering is used below.



The binary point is to the right of the low-order bit position 0. The 11-bit exponent carries a bias of 2000_8. Negative numbers are represented in one's complement notation, and exponent arithmetic is done in one's complement notation. The following table lists the normalized representation of various floating point numbers as octal bit patterns.

Table 1. Normalized Representation of Floating Point Numbers As Octal Bit Patterns

Number	Sign/Exponent	Mantissa
0.0	0000	0000 0000 0000 0000
1.0	1720	4000 0000 0000 0000
2.0	1721	4000 0000 0000 0000
-2.0	6056	3777 7777 7777 7777
5.0K226	2265	5000 0000 0000 0000
-5.0K226	5512	2777 7777 7777 7777
9.0K-80	1603	4400 0000 0000 0000
-9.0K-80	6174	3377 7777 7777 7777
indefinite	1777	any
-indefinite	5777	any
infinite	3777	any
-infinite	7777	any

Indefinite and infinite operands will cause the program to abort or propagate correctly depending on the option set with the mode card (User's Guide for the CDC 6700). If no mode card is used, then, by default, the program will abort when an infinite or indefinite operand is used, not when it is generated.

Arithmetic Precision of REAL Numbers. The precision for floating point numbers is one bit greater on the CDC 6700 than on the TDCC. The exponent is only 11 bits on the CDC as compared to 16 bits on the TDCC. This restricts the range of floating point numbers to:

$$.5E-293 \le x \le .5E321$$

Pointer Constants

The only legal pointer constant is 0. Of course, pointer expressions built up using THLL operators are valid at preset time and runtime. Pointer variables are 60-bit CDC integers. A pointer to a memory word is represented as the 18-bit offset of the memory word from the beginning of the control point. The pointer value is in the least significant 18 bits.

Boolean Constants

FALSE is represented as 0. TRUE is represented as an integer that has the low-order 32 bits set, but any non-zero value will test as true. If the value of TRUE is preset into a double variable, it is sign-extended to 60 bits. Since all 60 bits would be set, the double TRUE would test as 0 or FALSE. In CDC mode, doubles do not exist; therefore, the problem above cannot occur.

TRUE is different from -1 on the CDC 6700. On the BP, TRUE and -1 have the same bit pattern. TRICOMP6000 allows the questionable practice of using a generated TRUE/FALSE as an operand in a logical bit operation to work the same on the two machines.

Strings

A string is represented by a header word followed by a block of words holding the characters of a string in standard 6700 6-bit display code, packed 10 characters per word. When a string is assigned to a double variable, all 10 characters of the first word of the string body will be moved into the double. In CDC mode, a string can be assigned to an integer variable.

File names specified as strings in the second argument position of an OPEN call must be unique in the first seven characters. This restriction is imposed by SCOPE 3.4. Also, they must not contain dots.

DECLARATIONS

ALLOCATION OF STORAGE

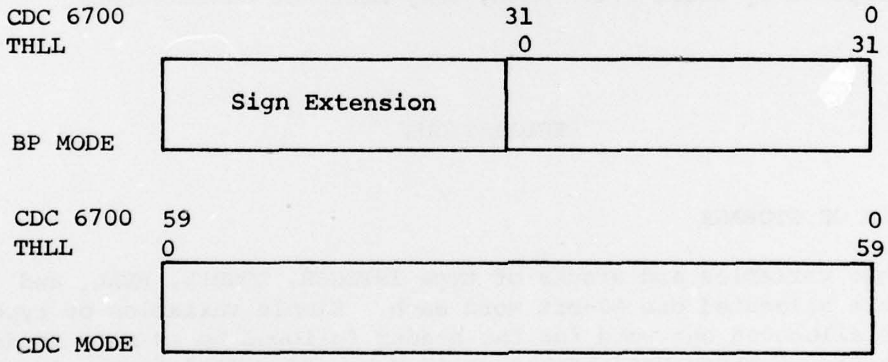
Simple variables and stacks of type INTEGER, DOUBLE, REAL, and POINTER are allocated one 60-bit word each. Simple variables of type ALPHA are allocated one word for the header followed by as many words as are needed to store N+1 characters if N was specified in the declaration.

HALF, INTEGER, and POINTER arrays are allocated one word per element. In CDC mode, REAL arrays are also one word per element. The DOUBLE attribute is changed to INTEGER attribute in CDC mode; therefore, DOUBLE arrays are interpreted as INTEGER arrays. In BP mode, DOUBLE and REAL arrays are allocated two words per element. In this case, only the first word is used for storing data values. The second word is not used and cannot be accessed using subscripted variables. This word, or parts of it, can be accessed using component variables and, since component offsets are always in units of words, this access method works the same way for the BP and the CDC for arrays of type INTEGER, DOUBLE, or REAL. Arrays of type HALF cannot be made to match up on both machines. Of course, since elements of a REAL or DOUBLE array are represented differently in the TDCC and CDC 6700, components of an array element could possibly yield different results on both machines. This programming practice is not machine independent.

If constants are stored in memory, then they are allocated storage the same way as simple variables of the corresponding type. Unused shared variables are not allocated storage on the runtime stack by TRICOMP6000.

BIT NUMBER CONVENTIONS

On the CDC 6700, the least significant bit or rightmost bit is numbered as bit 0 and the most significant or sign bit is numbered as bit 59. The TDCC numbers bits in the opposite order, thus bit 0 is the most significant bit and bit 31 is the least significant bit. Since THLL was designed originally for the TDCC, ascending bit numbers are from left to right.



In BP mode, bit 31 on the CDC 6700 is THLL bit 0, while in CDC mode, bit 59 on the CDC is THLL bit 0. Functions such as TEST.BIT, CLR.BIT, SET.BIT, TGL.BIT, and FIND.BIT use the THLL bit number as an input parameter, and FIND.BIT can also output a THLL bit number. These functions are implemented for both compiler modes and will, in general, yield different results since THLL bit assignments are different in both modes.

Indexed components also go from left to right in the order of increasing THLL bit assignments. In CDC mode, a 6-bit indexed component could be used to pick up CDC display code characters from THLL strings or ALPHA variables.

COMPONENTS

In BP mode, DOUBLE or REAL components or 32-bit wide INTEGER or POINTER components are considered full word components and all 60 bits of a CDC word will be referenced. In CDC mode only DOUBLE, REAL, and 60-bit wide INTEGER and POINTER components will access the full 60-bit CDC word. A 32-bit wide INTEGER component is precisely that in CDC mode, and it is a full word component in BP mode. If the field of a component is not specified, a full word component is assumed in either mode.

INTEGER and POINTER components generally have field specifications defining the start bit number and bit width of the component. The start bit number is the leftmost THLL bit number in that component. Since the compiler mode affects the THLL bit number mapping to CDC 6700 bits, care must be taken. In BP mode, the upper portion of a CDC word can be accessed by using full word components along with shifts and masking. The restriction on shifts is discussed later in this report.

The predefined components WHOLEW, DOUBLEW, and REALDW are always full 60-bit components.

Indexed Components

The behavior of indexed components depends on the compiler mode. When a component is indexed, the index is in units of fields. Full word INTEGER and POINTER components therefore index by words. In CDC mode, DOUBLE and REAL components also index as full word components. In BP mode, DOUBLE and REAL components index in units of two words while the field allocated for the component is the first of the two words.

Partial word INTEGER and POINTER components also depend upon the word width for that compiler mode. In CDC mode, the whole 60-bit CDC word is available for indexed components, while in BP mode, only the lower 32 bits of a CDC word are used for indexed components. The field width of a component to be indexed must divide exactly into the word width for that compiler mode, and the start bit position must be chosen as a multiple of the field width (including bit 0). Therefore, a field width of 15 bits is valid only for CDC mode, a field width of 16 bits is valid only for BP mode, and a 4-bit field width is valid for both modes.

PRESETS

The evaluation of compile time expressions is done using the arithmetic of the CDC computer, not by simulating the arithmetic of the TDCC. TRICOMP evaluates expressions as the TDCC does at runtime and TRICOMP6000 evaluates expressions as the CDC 6000 does at runtime. Therefore, each compiler evaluates compile time expressions in the environment of the target computer. The values produced can be different. TRICOMP6000 is subject to the restrictions on the arithmetic precision that were noted earlier in this report.

When a number is preset into a component, the CDC representation is placed into that component (X'FFFE' in TRICOMP6000 and X'FFFF' in TRICOMP represent -1 in a 16-bit field).

The preset functions LINKWORD and INITWORD have no purpose in the CDC 6700. They are recognized only in an effort to be compatible with TRICOMP for the TDCC.

COMMONS

TRICOMP6000 allows 63 COMMON declarations. This restriction does not exist in TRICOMP.

The allocation of storage within a common is the same as the allocation of storage outside of the common.

DIRECTIVES

The following directives serve no purpose in TRICOMP6000, but they are recognized tc maintain compatibility with TRICOMP for the TDCC: PRIV, NOPRIV, SCHEMA, NOSCHEMA, CRET, NOCRET, GDDF, NOGDDF, T, BIN, TMPMAX, OWNBR, INSBR, and CONBR.

TRICOMP6000 has two additional directives:

1. BP - compile in BP mode (32-bit words), and

2. CDC - compile in CDC mode (60-bit words).

The new directives must occur before the first BEGIN in a compile unit. An error message will be emitted and the directive ignored if this condition is not true. The default mode for TRICOMP6000 is the BP mode.

CONTROL CARD OPTIONS

The following control card options are in TRICOMP6000 only for the purpose of maintaining compatibility with TRICOMP for the TDCC: BIN, CONBR, CRET, EXEC, GDDF, INSBR, MAIN, OWNBR, SCHEMA, T, and TMPMAX. Also, the following control card option has been added to TRICOMP6000:

CDC - All compile units are to be compiled in CDC mode. The default mode is the BP mode. Directives can be used to override the selection of the compiler mode on the TRICOMP control card.

EXPRESSIONS AND STATEMENTS

Double constants and double variables are suppressed in CDC mode and are considered to be INTEGER; therefore, error recovery and error messages can be dependent on the compiler mode.

LINK AND EXEC PROCEDURES

LINK and EXEC procedures are assumed to be GLOBAL. It is not considered an error if they are also declared to be GLOBAL. Any global procedure can be used as the entry point for executing the object program.

RESTRICTIONS ON RECURSION

Procedures may be called in a recursive manner; that is, a procedure may call itself or call another procedure that will eventually call upon it. Parameters passed to a recursive procedure via the call line are safe only when passed by value. Since ALPHA variables, arrays, stacks, devices, formats, and entry points of procedures cannot be passed by value, they should not be used in calls to recursive procedures. When a parameter is passed by value, any change to a parameter while within the procedure will not affect the original parameter in the calling procedure. Therefore, recursive procedures cannot expect to return values through the procedure parameters. Since recursive procedures may have type INTEGER, DOUBLE, REAL, or POINTER, a single output can be returned as the procedure value. Additionally, global variables could be used to transmit results from a recursive procedure. In this case the same variable is used by each invocation of the recursive procedure.

THLL STANDARD FUNCTIONS

All standard functions as described in Section 3.3 of the THLL User's Guide (Reference 1) are available on the CDC 6700. DCP and RTOS service procedures are not available.

Most of the standard functions and a number of operators such as ** are implemented as routines in the runtime system consisting of the libraries THLLNK, AEDLNK, and FORTRAN.

The mathematical routines such as the trigonometric functions, SQRT, LN, and EXP are in FORTRAN. For the trigonometric functions no check is made to determine if the argument is in the range $-\pi \le x \le \pi$, and these functions are defined for all real numbers.

** behaves as on the TDCC except if the base and the exponent are 0:

$$\left.\begin{array}{l} 0\ \ \texttt{**}0 \\ 0.0\texttt{**}0 \end{array}\right\} \text{ is undefined, error message}$$

Shifting works as on the TDCC with the exception that SHIFTA(x,n) where n > 0 is equivalent to SHIFTR(x,n). This guarantees the following desirable properties of SHIFTA:

1. $\text{SHIFTA}(x,n) = \underbrace{\text{SHIFTA}(\ldots(\text{SHIFTA}}_{n \text{ times}},1)\ldots,1)\quad , n > 0$

2. If x and SHIFTA(x,1) have the same sign, then SHIFTA(x,1) = 2x.

The shift instructions work on the full 60-bit CDC word in both modes. Therefore, left-circular shifts could yield different results on the TDCC and CDC 6700.

The bit functions, TEST.BIT, CLR.BIT, SET.BIT, TGL.BIT, and FIND.BIT, work on the lower 32 bits of a CDC word in BP mode and the full 60-bit CDC word in CDC mode.

The string function ORDERC compares two strings and determines a relational value as its output (-1 for less, 0 for equal, 1 for greater). The rules for determining which string is less than or greater than the other are based on the ordering of characters on the TDCC. The arguments for ORDERC are in CDC display code, but the relation between the argument strings is determined based on the ASCII equivalent of those strings.

The SWA function is used only as a means of changing the type of a number from INTEGER to POINTER.

THLL UTILITY ROUTINES

The only utility routines documented in Appendix J of the THLL User's Guide (Reference 1) which are supported by the CDC runtime are TRUNC and TRUNCD.

SPECIAL UTILITY ROUTINES

Utility routines must be declared as externals to user programs. They can be found in the libraries made available to the TRICOMP6000 users: THLLNK, AEDLNK, and FORTRAN. The special utility routines are highly dependent on the AED library routines found in AEDLNK. Some of the following routines use an AED .C. string pointer which is actually a pointer to a string in AED format.

Routine DCOFTHLL will generate an AED string containing the same text as a THLL string. Each time DCOFTHLL is called, a new string is formed in the AED free-storage area.

P = DCOFTHLL(S) where:

S is a String

P is an AED .C. string pointer.

Routine OVLDGO is supplied to allow overlays to be loaded and immediately executed. This routine is supplied because the similar routine in the RTOS service procedures is not supported. OVLDGO is fully described in Section 7.4.2 of the AED User's Guide for the CDC 6000 (Reference 2).

OVLDGO (N1, N2, P) where:

N1 is primary overlay number as in an OVERLAY directive

N2 is secondary overlay number as in an OVERLAY directive

P is an AED .C. string pointer naming the local file name, lfn.

THLLERR and DAYFIL are routines that print messages in the dayfile of the job. THLLERR also causes that job to abort.

THLLERR(P) where:

P is an AED .C. string pointer.

DAYFIL(P) where:

P is an AED .C. string pointer.

EPRINT and TERM will print THLL strings on a single line on the output file in the indicated format.

EPRINT(S) where:

S is a string

EPRINT will print ****ERROR AT LABEL S in the system output file.

TERM(S) where:

S is a string

TERM will print S in the system output file.

ASMTHLL can be used along with other routines in the AED ASEMBL package, such as ASMDEC, CARET, etc., to construct lines in an output file (Chapter 4 in the AED Programmer's Guide, Reference 3).

P = ASMTHLL(S) where:

S is a string

P is the position in the line after insertion of the string S.

GETCH and PUTCH are routines that translate a character from the 6-bit display code used in TRICOMP6000 strings to the ASCII code used in TDCC strings, and vice versa.

I = GETCH(A,N) where:

A is an alpha variable or constant

N is an index into A $(0 \leq N < \text{LENGTH}(A))$

I is the ASCII code for the N^{th} position of A (after converting the character from 6-bit display code). If N is outside of the above range then I will be set to -1.

I = PUTCH(A,N,J) where:

A is an Alpha variable

N is an index into A as described above

J is an ASCII character code to be stored into the Nth position of A (after converting the character to 6-bit display code)

I equals J if N is in range, otherwise I is set to -1.

INFINITE and INDEFINITE are routines that can be used to check whether or not a real variable has these values. The CDC will cause an error mode whenever these values are used, not when they are created by some operation. These routines allow a program to take appropriate action instead of causing an error mode.

B = INFINITE(R) where:

R is a real value

B is set to TRUE if R is the CDC 6700 value "infinite".

B = INDEFINITE(R) where:

R is a real value

B is set to TRUE if R is the CDC 6700 value "indefinite".

I-O SUPPORT

The following points should be kept in mind by the THLL programmer using I-O on the CDC 6700.

The file name given in the call to the OPEN built-in function is interpreted as a SCOPE logical file name (LFN). The THLL file name must be unique in the first seven characters. Two files on different THLL devices cannot have the same LFN. The user must establish the correspondence between the LFN and the permanent file name, as with other languages running under SCOPE, by proper use of the ATTACH and CATALOG control cards.

Files on THLL devices CPRINT, SPRINT, and KBDSS are sequential. If the LFN used in the OPEN call is #'', the name #'INPUT' is used for input operations and #'OUTPUT' is used for output operations. Only writing is allowed for CPRINT and SPRINT; both reading and writing are allowed for KBDSS.

Files on THLL devices MDF, MTF, and ICL are random. The LFN used in an OPEN of a random file must not be #''.

Only formatted operations are allowed on sequential files. Only unformatted operations are permitted on random files.

The size of a formatted line for a sequential file is 4 * size given in the OPEN call. Thus a size of 20 should be used to read 80 column card images and 33 should be specified for 132 column printed output. The maximum input line size is 160 characters and the maximum output line size is 132 characters.

The output formatting conventions used on the CDC 6700 are slightly different from those used on the TDCC.

1. If the exponent of a REAL is in the range -99 to +99, only two digits are used for the exponent in E format and an extra digit of precision is used.

2. If a number to be printed in F format cannot be represented, E format is used instead.

3. I format output is accurate only to 48 bits, about 14 decimal digits H, O, and D formats are accurate to 60 bits.

4. If a REAL number is infinite, the output field is filled with "*".

5. If a REAL number is indefinite, the output field is filled with "?".

6. Column 1 of output line is interpreted as carriage control using standard CDC FORTRAN conventions.

Repeats can be nested ten deep.

If an attempt is made to read a non-existent record of a random file, the user's buffer is filled with 0.

A random file read by a THLL program must have been written by another THLL program.

Files must be closed. If an output file is not closed, some information may be lost when the THLL program finishes.

ERROR MESSAGES

COMPILER ERROR MESSAGES

The THLL User's Guide describes the error messages generated by REV 4 TRICOMP. The error messages from TRICOMP6000 can be different due to a number of reasons:

1. TRICOMP6000 is REV 3 and has a different pass 1 and pass 2,

2. The code generators are different due to different target machines, and

3. BP mode/CDC mode can cause differences in error messages.

Many of the error messages have the same error number in both compilers and the additional information (Appendix D in the THLL User's Guide, Reference 1) also applies to TRICOMP6000.

RUNTIME ERROR MESSAGES

Errors detected by the runtime system cause one of the following error messages to be placed in the job dayfile and the job to be terminated.

ARRAY SUBSCRIPT ERROR

An array subscript value is negative or larger than the declared bound.

CANNOT READ AFTER WRITE

Must rewind sequential file to read after write.

INCORRECT RECORD SIZE

Record size given in OPEN does not match record size of random file.

INPUT RECORD TOO LARGE

Maximum input line from sequential file is 160 characters which corresponds to OPEN(...,...,40).

INVALID FILE POSITION

Position given in random read or write is negative.

NO FORMAT GIVEN

Read or write on sequential file requires a format.

POLAR ANGLE OUT OF RANGE

The angle argument in the POCA function must be in the range -PI to +PI, inclusive.

RANDOM FILE REQUIRED

File name refers to a file not created by a THLL WRITE to a random file.

TOO MANY ELEMENTS POPPED FROM STACK

POP was attempted on an empty stack.

TOO MANY ELEMENTS PUSHED IN STACK

PUSH was attempted on full stack.

MAX REPEAT DEPTH EXCEEDED

Format repeat nesting exceeds 10.

CANNOT READ FROM FILE

File is not a legal read file. CPRINT and SPRINT are write only devices.

RANDOM FILE REQUIRES A NAME

The LFN used in an OPEN for a random file must not be #''.

EXECUTION OF TRICOMP6000

PROCEDURE LINKAGE

TRICOMP6000 uses the AED linkage conventions and linkage routines from the AED library. This means that AED and FORTRAN programs can be called from a THLL program and vice versa. In general, this is desirable and provides great flexibility. However, occasionally this has the unpleasant effect that satisfaction of externals by the loader can introduce duplicate names.

THLL and AED procedure names are truncated to six characters. FORTRAN procedure (subroutine) names must not exceed six characters if it is called from within a THLL or AED procedure. AED procedures can have names that are not legal THLL identifiers. .S.P.M is an AED library procedure that is not callable from a THLL procedure. Such a procedure could be called by writing an interface procedure in AED in the following manner:

```
SPM
BEGIN
PROCEDURE .S.P.M;
DEFINE PROCEDURE SPM (MEMSIZE) WHERE
     POINTER MEMSIZE
     TOBE
     .S.P.M (MEMSIZE);
END FINI
```

A THLL procedure can call procedure SPM, which in turn calls .S.P.M. Procedure .S.P.M is useful in a multiple-overlay situation. It informs the AED free-storage package that MEMSIZE is above the largest overlay and, therefore, free-storage allocation is safe above MEMSIZE. The AED free-storage package is used by THLL programs to handle the runtime stack. Procedure .S.P.M must be the first procedure call from the main program. The loader field length is used by the free-storage package for a program that has no overlays. Therefore, a call to procedure .S.P.M is not necessary. (See Section 7.3.3 in the AED User's Guide, Reference 2, and Chapter 2 in the AED Programmer's Guide, Reference 3, for more information about the AED free-storage package.)

AED and FORTRAN arrays have no header, but THLL arrays do. This presents a problem in communicating via array names. As a rule, an array name should not be made EXTERNAL or passed as an argument to a procedure across languages, say between AED and THLL. A pointer to the origin of the data in the array should be used instead.

There is no BINDER for the CDC 6700. Therefore, there is no compatibility checking of GLOBAL/EXTERNAL symbols.

CONTROL CARDS

Control card examples are given in Appendix A. The following gives a brief description of the control card sequences and their function.

```
ATTACH,INSERTS,...
ATTACH,TRICOMP6000,ID=NCL.
TRICOMP,INPUT,OUTPUT.
```

will attach a user insert file, attach, and execute the compiler with source code on system input, the source listing on system output, and the COMPASS assembly code on file BPCODE.

```
REWIND,BPCODE.
COMBINE,BPCODE,ONEINP,999.
REWIND,BPCODE,ONEINP.
```

will combine the COMPASS assembly source into one zero-level record if the user's source input contains more than one compile unit. The combine and second rewind are not necessary if one compile unit is supplied.

```
ATTACH,THLLTXT,ID=NCL.
REQUEST,BIN,*PF.
COMPASS,I=BPCODE,G=THLLTXT,B=BIN,L=O.
```

will attach the THLL macro file and execute the COMPASS assembler. Source input is file BPCODE, the macro file is THLLTXT, the binary is permanent file BIN, and the COMPASS listing is suppressed.

```
ATTACH,THLLNK,ID=NCL.
ATTACH,AEDLNK,ID=ND4.
LDSET,LIB=THLLNK/AEDLNK/FORTRAN.
LOAD,BIN.
EXECUTE,THLLJOB.
```

will attach the THLL and AED runtime support and define them as system libraries with the FORTRAN library. File BIN will be loaded and executed with the program entry point being the Global Procedure THLLJOB.

```
ATTACH,OLDPL,...
REQUEST,NEWPL,*PF.
UPDATE,N,C.
CATALOG,NEWPL,...
PURGE,OLDPL.
RENAME,NEWPL,,CY=1.
RETURN,OLDPL,NEWPL.
.
.
.
7-8-9
Update Correction Deck
7-8-9
```

is a typical update sequence that will update an existing user PL (CY=1 in this example), create a compile file, catalog a new PL as cycle 2, purge the old PL, and rename the new PL as cycle 1.

```
ATTACH,OLD,RELBIN,...
REQUEST,NEW,*PF.
REWIND,OLD,MOD,NEW.
COPYL,OLD,MOD,NEW,,A.
CATALOG,NEW,RELBIN,...
PURGE,OLD.
RENAME,NEW,,CY=1.
```

is a typical COPYL sequence that will update an existing user relocatable binary file (CY=1 in this example), catalog the new RELBIN as cycle 2, purge the old RELBIN, and rename the new RELBIN cycle 1. File MOD contains the binary information to be replaced on RELBIN. COPYL,OLD,MOD, NEW,,A will add new binaries as well as replace existing binaries.

Examples 4 and 5 in Appendix A are typical examples of creating and executing an absolute binary file. An absolute binary file contains all of the required information to execute, i.e., no system or user libraries are needed at runtime. The COMPASS routine LINK provides the necessary entry point sequence for the CYBER Loader. TRICOMP6000 does not provide this information. If a binary contains more than one overlay, then each overlay must contain its own COMPASS routine for entry point names.

REFERENCES

1. *TRIDENT Higher Level Language User's Guide*, The THLL Group, Warfare Analysis Department, July 1977 (Revised June 1978), Naval Surface Weapons Center, NSWC/DL TR-3657.

2. *AED User's Guide*, CDC 6000 Edition, September 1973, SofTech, Inc., Waltham, Mass.

3. *AED Programmer's Guide*, August 1973, SofTech, Inc., Waltham, Mass.

B028740

APPENDIX A

CONTROL CARD EXAMPLES

TRICOMP6000 COMPILE AND CATALOG

```
Job Card
Account Card
ATTACH,TRICOMP6000,ID=NCL.
TRICOMP,INPUT,OUTPUT.
REWIND,BPCODE.
COMBINE,BPCODE,ONEINP,999.
REWIND,BPCODE,ONEINP.
ATTACH,THLLTXT,ID=NCL.
REQUEST,BIN,*PF.
COMPASS,I=ONEINP,G=THLLTXT,B=BIN,L=0.
RETURN,BPCODE,ONEINP,THLLTXT.
CATALOG,BIN,...
7-8-9
THLLJOB
BEGIN
   GLOBAL THLLJOB;
   DEFINE PROCEDURE THLLJOB;
      BEGIN
      .
      .
      .
      END;
END
FINIS
7-8-9
6-7-8-9
```

Control Card - Example 1

TRICOMP6000 COMPILE FROM UPDATE PL AND COPYL TO RELBIN

```
Job Card
Account Card
ATTACH,OLDPL,...
REQUEST,NEWPL,*PF.
UPDATE,N,C.
CATALOG,NEWPL,...
PURGE,OLDPL.
RENAME,NEWPL,,CY=1.
RETURN,OLDPL,NEWPL.
ATTACH,INSERTS,...
ATTACH,TRICOMP6000,ID=NCL.
TRICOMP,COMPILE,OUTPUT.
REWIND,BPCODE.
COMBINE,BPCODE,ONEINP,999.
REWIND,BPCODE,ONEINP.
ATTACH,THLLTXT,ID=NCL.
COMPASS,I=ONEINP,G=THLLTXT,B=MOD,L=0.
RETURN,BPCODE,ONEINP,THLLTXT.
ATTACH,OLD,RELBIN,...
REQUEST,NEW,*PF.
REWIND,OLD,MOD,NEW.
COPYL,OLD,MOD,NEW,,A.
CATALOG,NEW,RELBIN,...
PURGE,OLD.
RENAME,NEW,,CY=1.
7-8-9
Update correction deck
7-8-9
6-7-8-9
```

Note: COPYL,OLD,MOD,NEW,,A will add new binaries as well as replace existing binaries.

Control Card - Example 2

TRICOMP6000 EXECUTE FROM RELBIN

```
Job Card
Account Card
ATTACH,RELBIN,...
ATTACH,THLLNK,ID=NCL.
ATTACH,AEDLNK,ID=ND4.
LDSET,LIB=THLLNK/AEDLNK/FORTRAN.
LOAD,RELBIN.
EXECUTE,THLLJOB.
7-8-9
6-7-8-9
```

Note: THLLJOB is the Global or Link procedure name for this example.

Control Card - Example 3

TRICOMP6000 NOGO AND CATALOG ABSBIN

```
Job Card
Account Card
COMPASS,B=LINK,L=0.
ATTACH,RELBIN,...
REQUEST,ABSBIN,*PF.
MAP,ON.
ATTACH,THLLNK,ID=NCL.
ATTACH,AEDLNK,ID=ND4.
LDSET,LIB=THLLNK/AEDLNK/FORTRAN.
LOAD,RELBIN,LINK.
NOGO,ABSBIN.
CATALOG,ABSBIN,...
7-8-9
          IDENT LINK
          TITLE LINK      OVERLAY(0,0)ENTRY
          EXT   MAIN
          END   MAIN
7-8-9
6-7-8-9
```

Note: MAIN is the Global or Link procedure name for this example.

Control Card - Example 4

TRICOMP6000 EXECUTE FROM ABSBIN

```
Job Card
Account Card
ATTACH,ABSBIN,...
ABSBIN.
7-8-9
6-7-8-9
```

Control Card - Example 5

TRICOMP6000 COMPILE AND EXECUTE

```
Job Card
Account Card
ATTACH,TRICOMP6000,ID=NCL.
TRICOMP,INPUT,OUTPUT.
REWIND,BPCODE.
COMBINE,BPCODE,ONEINP,999.
REWIND,BPCODE,ONEINP.
ATTACH,THLLTXT,ID=NCL.
COMPASS,I=ONEINP,G=THLLTXT,B=BIN,L=0.
RETURN,BPCODE,ONEINP,THLLTXT.
ATTACH,THLLNK,ID=NCL.
ATTACH,AEDLNK,ID=ND4.
LDSET,LIB=THLLNK/AEDLNK/FORTRAN.
LOAD,BIN.
EXECUTE,THLLJOB.
7-8-9
THLLJOB
BEGIN
     GLOBAL THLLJOB;
     DEFINE LINK PROCEDURE THLLJOB;
          BEGIN
          .
          .
          .
          END;
END
FINIS
7-8-9
6-7-8-9
```

Control Card - Example 6

DISTRIBUTION

General Electric Company
Ordnance Systems
100 Plastics Avenue
Pittsfield, MA 21201
Attn: E. Zeller

Strategic Systems Project Office
Department of the Navy
Washington, DC 20376
Attn: SP23115 (C. Chappell)

EG&G Washington Analytical Services Center
P.O. Box 552
Dahlgren, VA 22448
Attn: S. Franzello (2)
Library

Defense Documentation Center
Cameron Station
Alexandria, VA 22314 (2)

Local

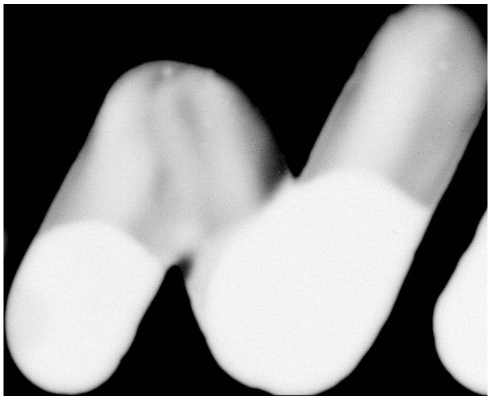
K50-GE	
K51	2
K53	2
K54	50
K56	2
X210	2
E41	

END

DATE
FILMED

0-79

DDC



AD-A073 904   NAVAL SURFACE WEAPONS CENTER DAHLGREN LAB VA   F/G 9/2
TRICOMP6000 USER'S GUIDE.(U)
MAR 79 W CARUTHERS, H HUBER, J ZALOUDEK
UNCLASSIFIED NSWC/TR-3938   NL

2 OF 2
AD A
073904

SUPPLEMENTARY

INFORMATION

END
DATE
FILMED
4-81
DTIC

D A
073

SUPPLEMENTARY

INFORMATION

AD-A073904

DEPARTMENT OF THE NAVY
NAVAL SURFACE WEAPONS CENTER
DAHLGREN, VIRGINIA 22448

WHITE OAK LABORATORY
SILVER SPRING, MD. 20910
(202) 394-

DAHLGREN LABORATORY
DAHLGREN, VA. 22448
(703) 663-

IN REPLY REFER TO
8656

UNCLASSIFIED

To All Holders of NSWC/DL TR-3938

DATE: 1 March 1981

Insert Change 1

R. T. RYLAND, JR.
By direction

NSWC/DL TR-3938
Dated 1 March 1979
Is Changed as Follows:

TITLE: TRICOMP6000 USER'S GUIDE

1. Remove pages 13 and 14 and insert new pages 13 and 14.

2. Insert this letter between front cover and Report Documentation page.

N is an index into A as described above

J is an ASCII character code to be stored into the Nth position of A (after converting the character to 6-bit display code)

I equals J if N is in range, otherwise I is set to -1.

INFINITE and INDEFINITE are routines that can be used to check whether or not a real variable has these values. The CDC will cause an error mode whenever these values are used, not when they are created by some operation. These routines allow a program to take appropriate action instead of causing an error mode.

B = INFINITE(R) where:

R is a real value

B is set to TRUE if R is the CDC 6700 value "infinite".

B = INDEFINITE(R) where:

R is a real value

B is set to TRUE if R is the CDC 6700 value "indefinite".

I-O SUPPORT

The following points should be kept in mind by the THLL programmer using I-O on the CDC 6700.

The file name given in the call to the OPEN built-in function is interpreted as a SCOPE logical file name (LFN). The THLL file name must be unique in the first seven characters. Two files on different THLL devices cannot have the same LFN. The user must establish the correspondence between the LFN and the permanent file name, as with other languages running under SCOPE, by proper use of the ATTACH and CATALOG control cards.

Files on THLL devices CPRINT, SPRINT, and KBDSS are sequential. If the LFN used in the OPEN call is #'', the name #'INPUT' is used for input operations and #'OUTPUT' is used for output operations. Only writing is allowed for CPRINT and SPRINT; both reading and writing are allowed for KBDSS.

Files on THLL devices MDF, MTF, and ICL are random. The LFN used in an OPEN of a random file must not be #''.

Only formatted operations are allowed on sequential files. Only unformatted operations are permitted on random files.

The size of a formatted line for a sequential file is 4 * size given in the OPEN call. Thus a size of 20 should be used to read 80 column card images and 33 should be specified for 132 column printed output. The maximum input line size is 160 characters and the maximum output line size is 132 characters.

The output formatting conventions used on the CDC 6700 are slightly different from those used on the TDCC.

1. If the exponent of a REAL is in the range -99 to +99, only two digits are used for the exponent in E format and an extra digit of precision is used.

2. If a number to be printed in F format cannot be represented, E format is used instead.

3. I format output is accurate only to 48 bits, about 14 decimal digits. H, O, and D formats are accurate to 60 bits.

4. If a REAL number is infinite, the output field is filled with "*".

5. If a REAL number is indefinite, the output field is filled with "?".

6. For files other than system file OUTPUT, column 1 of the output line is interpreted as the carriage control using standard CDC FORTRAN conventions. System file OUTPUT will have a blank space control appended to the output line.

Repeats can be nested ten deep.

If an attempt is made to read a non-existent record of a random file, the user's buffer is filled with 0.

A random file read by a THLL program must have been written by another THLL program.

Files must be closed. If an output file is not closed, some information may be lost when the THLL program finishes.